Clean and marked copies of the pages of the specification affected by the above-described changes are submitted herewith.

IN THE CLAIMS

Cancel claims 1-10.

In claim 15 change the reference to "claim 13" to --claim 14--.

Clean and marked copies of the claims as amended are provided herewith.

REMARKS

Applicant thanks the Examiner for his review of the application.

Claims 1-19 are in the application. In the Office Action, the Examiner:

- Objected to the specification and the claims and required correction of certain informalities.
- Noted that certain IDS references had not been considered because they were not found in the file.
- Rejected claims 1-15 under Section 102 (b) based on public use of a system of applicant's called "KAPITAL" as described in an article in Sloan Management Review of Winter 1995 by Richard Pawson, et al. ("Pawson").
- Rejected claims 1-10 under Section 102 (b) as anticipated by an article by Eggenschwiller et al. in Conference on Object Oriented Programming Systems, Languages and Applications of October 1992 ("Eggenschwiller").
- Rejected claims 11-15 under Section 103 (a) over Kleckner et al, WO94120912 ("Kleckner") in view of Rasala, "A model C++ tree iterator class for binary search trees", in 28th SIGCSE Technical Symposium on Computer Science Education in March 1997 ("Rasala").

- Rejected claims 16-19 under Section 103 (a) over Kleckner and Rasala as applied to claim 11, further in view of Gould, "Double Dispatch with an Inverted Visitor Pattern," in the May 1998 edition of C/C++ Users Journal ("Gould").

Summary of Response. In response to the Office Action, Applicant:

- Corrects informalities in the specification and the claims, including starting the claims on a separate page.
- Corrects a typographical error in claim 15.
- Submits an explanation as to missing references (which applicant states were in fact submitted), and submits additional copies thereof, with a request that the Examiner consider the same.
- Cancels claims 1-10.
- Traverses the public use rejection insofar as directed to the non-canceled claims (11-15).
- Traverses the rejection of claims 11-15 over Kleckner and Rasala.
- Traverses the rejection of claims 16-19 over Kleckner, Rasala and Gould.

As hereby amended, because of the cancellation of claims 1-10, only claims 11-19 remain before the Examiner for consideration at this time.

Detailed Remarks

IDS Reference Copies. Applicant submitted copies of all references cited in its Information Disclosure Statement submitted on April 1, 1999. The Examiner may not have noted, however, that there were *two* submissions in this regard. The first submission contained copies of all cited references. However, with respect to the Goldberg et al. reference, the undersigned realized after sending it, that it comprised over 700 pages, and that only a few pages of the reference were in fact relevant. Accordingly, applicant sent a follow-up letter (also on April 1, 1999) that noted the bulk of the Goldberg et al. reference and submitted separately the excerpted relevant pages of that reference. It may be that the Examiner picked up the second

submission (the follow-up letter), found the excerpted Goldberg et al. materials, and did not realize that an earlier submission actually contained copies of all of the other references. A complete set of all references should be in the file, together with the original full copy of the bulky Goldberg et al. reference.

The undersigned represents from personal knowledge and present recollection that copies of all cited references went into the envelope with the original IDS that was submitted to the Office. For the convenience of the Examiner, an additional full set of copies has been submitted to the Examiner via electronic mail, pursuant to the authorization for Internet Email recently entered in the file. Applicant requests that all cited references that have not as yet been considered be considered and made of record, and that the IDS be marked to so indicate.

Public Use Rejection of Claims 11-15. Applicant's "KAPITAL" system as implemented prior to July 31, 1997 "critical date" (the date 12 months prior to the filing date of this application) was concerned with using object oriented methods to construct financial instruments as software "objects" and manipulating those objects.

The Kapital system as developed and used by applicant prior to the July 31, 1997 critical date of the present application concerned a system with its origins in the early 1990s for using object oriented software, written primarily in the Smalltalk language, to model financial derivative instruments. The foremost objectives of the system during that period concerned providing an easy-to-use mechanism for describing and modeling diverse instruments of the sort encountered in connection with derivatives trading. The object models employed during these stages of development were conventional object-oriented models wherein the processes involved in creating and managing the instruments were encapsulated in each instrument. This included processing operations such as pricing and valuation.

After the critical date, applicant's development efforts turned to the question of how to manage a portfolio of complex financial instruments that have been developed in an object oriented system, for example, how to value the diverse instruments in the portfolio in a consistent manner, so as to be able to have reliable and meaningful estimates of the value of the

entire portfolio at the desired times. This resulted in further development, including without limitation the development of the concepts of converting financial instrument objects into "financial event streams" and devising separate "processing objects" to process those streams so as to perform crucial tasks, such as pricing and valuation. Pricing and valuation must be performed for a plurality of diverse instruments held by a financial institution, and most desirably should be performed in a consistent manner, so as to render reliable and reproducible pricing/valuation figures representing the institution's actual financial position as a result of holding the portfolio. Applicant's post-critical date efforts in this regard concerned building the pricing, valuation and other processing functions into "processing objects" separate from the underlying financial instrument objects themselves, reflecting an architectural change from the earlier versions of the system that had been in use. It is this revised, post-critical date architecture that is the subject of the present application.

The concept of a "processing object" distinct from "financial instrument objects," and the motivation for separating processing from instrument construction and representation, is discussed throughout the specification, and is treated intensively in the discussion that begins on line 24 of page 39. The overall architecture of a system employing such separate processing objects is shown in Figure 7 (in particular, under the headings "Process Instruments" and "Stream Processing Objects").

Claims 11-15 (as well as 16-19) specifically concern those issues that arise in the course of managing a plurality of complex financial instruments, and reflect the unique system architecture that resulted from an approach based on having separate "processing objects" to perform significant processing such as pricing and valuation.

Claims 11-15 all contain the base limitations of claim 11, including applying a generic traversal process to the macro structure of a financial instrument to implement one or more functions that produce results based on the information developed from such traversal. It is clear from the specification (see, e.g., p. 4, lines 8-12) that the "generic traversal process" referred to

in claim 11 is the minimalistic form of a "processing object," which gets further fleshed out as functionality is added to it in claims 12 – 15.

As will be discussed in further detail below, there are alternative approaches to processing a plurality of financial instruments in an object oriented system. The present approach involves using processing functions external to the financial instrument objects themselves. Being external, these functions do not initially "know" the properties of the instruments they are supposed to operate on (see spec. p. 43, line 25 – page 44, line 36: "the processor does not . . . have any a priori knowledge of the structure of a financial instrument. . . . This is a very important concept as it allows a processor to process any well formed macro structure because the macro structure itself drives the sequence of processing steps."). To acquire that information regarding the macro structure, a traversal process is employed. KAPITAL as implemented prior to the critical date did not process financial instruments in this manner. There was no "generic traversal process" in those versions of KAPITAL as contemplated by claims 11-15. Thus, the subject matter of claims 11-15 was not in public use prior to the critical date.

Rejection of claims 11-15 based on Kleckner and Rasala. Kleckner discloses an object oriented system for modeling and manipulating a financial instrument. Rasala discloses an iterator class for a binary tree, which iterator class can return the items in the binary tree in a specified order.

Claim 11 reads as follows:

11. A system comprising data processing means wherein a generic traversal process is employed that can be applied to the macro structure of a financial instrument to implement one or more functions that produce results based on this information.

Claims 11-15 concern techniques that are useful for implementing a "processing object" for processing financial instruments.

As noted above, the present invention uses functions of a processing object external to the individual financial instrument objects in order to conduct certain processing (such as

valuation) with regard to those objects. In claim 11, which is the base claim in this series, the processing object is stripped to its most minimalistic form, which is a generic traversal process.

In an object oriented system, an object such as a processing object cannot be presumed to know the structure of other objects such as financial instrument objects. The "macro structure" of a financial instrument, as explained at page 9, lines 5-15 of the specification, is reflected in properties that are internal to and "private" within the instrument. As illustrated in Fig. 3 of the present application, the macro structure is a higher level structure representing how the instrument's "micro structure" components are related or nested within each other. In the present invention, a traversal process is employed so as to expose to the external processing object (the "generic traversal process" of claim 11) the macro structure information for the instrument.

Kleckner in light of Rasala does not render claim 11 obvious.

First, Kleckner and Rasala in combination fail to teach all of the elements of claim 11 (and of dependent claims 12-15). Fundamentally, the financial instruments of Kleckner do not have a "macro structure." The instruments of Kleckner have financial terms, which correspond to the "micro structure" as described in the present application, rather than the "macro structure" recited in claim 11. Moreover, Rasala teaches a specific form of traversal for binary trees. It does not teach a generic traversal process. Furthermore, another explicit element of claim 11 is applying the "generic traversal process" to the macro structure of the instrument: in addition to the elements of traversal and macro structure both being present, this claim further expressly requires that the one element (traversal) be "applied" to the other (macro structure). Kleckner, by contrast, does not apply, and based on its teachings would derive no benefit from applying, a traversal process to any structural attributes of its instruments. Rasala addresses a specific type of traversal, but does not teach applying traversal to anything other than binary trees.

Second, there is no motivation within the cited references or otherwise apparent, for combining them so as to arrive at what is recited in claims 11-15.

One of the governing principles in a classic object oriented system is "encapsulation" – every object contains not only its own data (properties), but the functions for accessing and

manipulating those properties. Kleckner fits the *conventional* encapsulation paradigm: if a result (such as valuation) is desired as to a financial instrument object, a function is provided internally to the object which can be called so that it will return the desired information. Neither Kleckner nor Rasala, alone or in combination, teach traversing the macro structure of the instrument, as opposed to the instrument itself, because in a pure encapsulation approach, there is no need ever to do so – each instrument has within it all necessary *internal* functions for processing internal data. These functions already "know" the structure of the instrument to which they belong (which in the case of Kleckner is a micro structure). Thus there is no need or use for a traversal process as recited in claims 11-15.

In taking an "encapsulated" approach to processing, Kleckner in fact strongly teaches away from the approach taken by claims 11-15. This may be seen in Fig. 8 of Kleckner, and in the corresponding discussion on page 20, lines 6-18, wherein the "net present value" of an instrument is derived by applying its internal "curve rate return" method. This is simply the conventional approach to valuation. The practical benefits resulting from the alternative approach followed by the present invention are described at page 52, lines 1-23 of the specification. Kleckner, while it develops good techniques for *creating* object oriented instruments, teaches only the conventional encapsulated approach, with all its drawbacks when it comes to processing those instruments. That approach is inconsistent with the approach taken by the present invention, which is to handle such processing with "processing objects" *external* to the financial instrument objects themselves. Kleckner, whose system architecture is based on encapsulation, would lead a person of ordinary skill in the art away from, rather than towards, the solution recited in claims 11-15 of the present invention.

Although claims 12-15 are all dependent from claim 11, and therefore all patentable for the same reasons as given above, applicant will briefly address the additional limitations of claims 12-15 which further distinguish those claims from the prior art.

Claim 12 reads as follows:

12. The system of claim 11, wherein each said function is implemented as a specific extension of said generic traversal process to generate a specified type of result.

Claim 12 concerns implementing the "function" recited in claim 11 as a specific extension of the generic traversal process to generate a specified type of result. In other words, the "function" becomes part of the "processing object" – the separate, external object that processes the information provided by the traversal. As noted above, Kleckner handles this processing internally, and thus has no need for any such traversal. Saying that the function is implemented as an "extension" to the traversal is completely foreign to anything taught or suggested by Kleckner (or Rasala).

Claim 13 reads as follows:

13. The system of claim 12, wherein each traversal process is based on a well defined interface between the financial events contained in the financial event structure of a financial instrument and said traversal process.

Claim 13 extends claims 12 and 11 in accordance with what lies behind the "macro" financial event "structure" of a financial instrument – i.e., the "micro structure" of the individual "financial events" underlying the instrument. In claim 13, the traversal process (processing object) is based on a well defined interface to the financial events out of which every financial instrument is constructed. In other words, this is a full componentized approach to both construction and processing, wherein the financial instrument objects are constructed out of certain components; a macro structure is created to reflect the interrelationship of those components; the processing objects traverse the macro structure in order to derive which components are present and how they are interrelated; and those processing objects have, and apply, a well defined interface to these components.

As explained in the specification, every financial instrument in the present invention is defined atomistically in terms of very simple "financial event" building blocks (payments, accruals, option exercises, etc) forming its "micro structure," and the higher-level structure by which those events are related is called the "macro" or "financial event structure." See Figures 2

and 3 and corresponding discussion at pages 7-9 of the specification. Conceptually, the "micro" and "macro" structures are two layers, both reflected in the actual structure of each financial instrument.

As explained above, claim 13 introduces the additional limitation of basing each traversal process as extended by the processing functions of claim 12 on a *well-defined interface* between (a) the financial events contained in the financial event structure of a financial instrument and (b) the traversal process itself. The Examiner's rejection does not address this limitation. In fact, the Examiner's rejection of claim 13 is verbatim identical to the rejection of claim 12, and adds nothing to it. In any event, the limitation in question is nowhere to be found in Kleckner or Rasala.

The Examiner has contended here and in connection with other claims in this application that the subject matter is "inherent" in the functionality of C++. It is respectfully submitted that this would confuse the capabilities of a tool (the C++ language) from the infinite variety of programs that can be constructed with the tool. To be sure, a great many techniques for writing C++ programs (and other object oriented programs) are well known, and of course many of them are packaged as extensions to the language in libraries included with standard compiler distribution packages. But the question here is whether the particular specifically claimed techniques, as applied to financial instrument objects, are novel and nonobvious. The cited references, Kleckner and Rasala, do not establish lack of novelty or obviousness; merely referring to the unspecified "inherent" capabilities of C++ adds no further support for this rejection.

Claim 14 reads as follows:

14. The system of claim 13, wherein the action to be performed for each type of financial event is defined, in said specific traversal process, independently from the action for any other type of financial event.

Claim 14 addresses the independence of each function of the processing object (the traversal process) directed at each financial event underlying an instrument, from the functions

therein directed at any other financial events. The idea of using financial event components is made much easier to implement if the components, and the functions that deal with them, are chosen and constructed so as not to have unintended interactions. In addition, because every financial event is processed independently, such independence facilitates arriving at coherent and consistent results when a plurality of financial instruments, containing different combinations of such underlying financial events, must be processed.

The Examiner, in response to claim 14, repeated the same rejection provided with respect to claims 12 and 13, adding merely, with respect to Kleckner, that "[i]ndependence is shown on page 4, line 39" (actually, the reference should be to page 4, line 31, as there is no line 39) . This reference in Kleckner is to the "term evaluation engine" of that invention. As can be readily seen, Kleckner's "term evaluation engine" is a functionality internal to and encapsulated within each financial object for evaluating the network of interconnected terms of a *single instrument*. According to Kleckner, this evaluation is "order independent for inputs" making it amenable to parallel processing. However, Kleckner's function definition takes place internal to the instrument. The present invention defines the operative functions and their interfaces within separate processing objects, which must be able to process inputs in an independent manner from whatever financial instrument it is directed at. The actions of Kleckner are not defined "in said specific traversal process," for there is no traversal process in Kleckner, and nothing in Kleckner teaches the desirability of consistent treatment from one financial instrument to the next, or any manner of achieving such consistency. Moreover, whereas Kleckner teaches operations that are order independent, in the present invention, the order of processing is determined by an instrument's macro structure (see specification, page 44, lines 18-19).

Claim 15 (as hereby amended) reads as follows:

15. The system of claim 14, wherein the overall result of applying a function specific traversal process to the financial event structure of a financial instrument is a combination of applying all individual financial actions to the respective financial events in a prescribed way.

Claim 15 has been amended. The parent claim reference in claim 15 should have been to claim 14 rather than claim 13. The error was merely typographical, and this can be seen from claim 15's reference to "actions", which term is only introduced per se in claim 14.

Claim 15 contains all the limitations of claims 11-14, and recites the overall result of applying a plurality of function-specific traversal processes to the financial event structure of a financial instrument as a combination of applying all the individual financial actions (i.e., the independent actions of claim 14) to the respective financial events of the instrument in a prescribed way. Claim 15 is directed to a property that results from applying the methodology of claims 11-14, referred to in mathematics as "distributivity." In other words, the system is constructed in such a way a processing object processes an instrument having a particular event structure so as to arrive at a result that is simply a combination of applying the individual functions of the processing object to the respective financial events of the instrument.

The rejection of this claim adds to the prior rejection paragraphs a reference to fig. 6 of Kleckner. That figure illustrates the handling of one instrument, an equity spread option. The "combination" of actions in this figure is the combination of actions specified in the internal processing function for the instrument. While that much is apparent, there is no teaching here about the invariance of processing the components individually as always yielding the same result as processing them as a group, and further, no suggestion to do this as part of a "traversal process" separate from the underlying instrument (i.e., in a manner that can be applied to a plurality of instruments constructed from the very same elements).

Rejection of claims 16-19 based on Kleckner and Rasala in view of Gould. Claims 16-19, like claims 12-15, are also dependent from claim 11, and therefore patentable for the same reasons given with respect to claim 11. Further bases for patentability specific to claims 16-19 are given below.

Claim 16 reads as follows:

16. The system of claim 11, wherein said traversal process is implemented via a double dispatch mechanism.

As discussed above, claim 11 concerns a processing object in the form of a "generic traversal process" that performs processing such as pricing and valuation on one or more financial instruments, each having "micro" and "macro" event structures. Claims 16-19 concern an implementation of the technique of claim 11 via a double-dispatch mechanism. In other words, while traversing the macro structure of the financial instrument, the processing object must determine and apply the proper processing methods to the financial events it encounters during the traversal. Double dispatch is one method that can be used to determine which processing method is chosen for a given financial event within a given processing object. It is the use of double dispatch within the traversal process of claim 11 that is claimed in claim 16.

In object oriented programming, "late binding" to a "virtual function" can be used to achieve "polymorphism" – to select an appropriate form of a method based on a change in a parameter in a target. This is called "single dispatch." A similar result can be obtained by "overloading" functions with other variants. However, when both the target and the processor are substitutable, single dispatch will not work. An object oriented work-around for this problem is "double dispatch," in which the target object is queried, and returns to the processor the proper classification of a function to invoke in order to effect the processing.

Double dispatch was of course well known and applicant of course does not claim to have invented it. But the application of double dispatch to solve the problems addressed by the present invention was neither anticipated nor rendered obvious by anything in the prior art.

The Examiner applied Gould in light of Kleckner and Rasala as applied to claim 11. For the reasons given above, Kleckner and Rasala do not obviate claim 11, so merely adding Gould (a double-dispatch reference) does not set up a valid basis for rejecting claim 16 (or claims 17-19 which also depend from claim 11).

Moreover, there is no suggestion or motivation to combine double dispatch as taught by Gould to the financial instrument modeling as taught by Kleckner and the iterators taught by Rasala. Such a suggestion or motiviation is essential in order to support a prima facie case of obviousness. MPEP 2142. In Kleckner, no necessity is apparent for having to resolve which

processing function to apply, in that the reference does not teach that there is a choice of functions – the reference does not even teach single dispatch. There would be no reason whatsoever in the context of Kleckner, in which (unlike the present invention) the processing functions already have knowledge of the instrument structure, to resort to double dispatch. There would be no need to do so, and no advantage in doing so.

The Examiner stated that a motivation for adding double dispatch as taught by Gould to Kleckner and Rasala would be to "allow for more elegant and therefore less costly programming." Double dispatch, however, with its back-and-forth calls and handshaking (see Fig. 11 of the present specification), would only have needlessly complicated Kleckner, where a simple method invocation is all that would appear to be necessary. No motivation at all is apparent from Kleckner to select double dispatch as a means of determining the proper function for processing financial events. Indeed, adding the unnecessary overhead of double dispatch to Kleckner would make its processing more costly rather than less costly.

Although claims 17-19 are all dependent from claim 16, and therefore all patentable for the same reasons as given above, applicant will briefly address the additional limitations of claims 17-19 which further distinguish those claims from the prior art.

Claim 17 reads as follows:

17. The system of claim 16, wherein said double dispatch mechanism selects the appropriate action for each financial event without predetermined knowledge of the overall referential structure of the financial event structure.

Claim 17 specifically recites that the double dispatch mechanism that implements the traversal process selects the appropriate action for each financial event it encounters without predetermined knowledge of the overall referential structure of the financial event structure. Kleckner teaches processing functions internal to the financial instrument object; there is no "traversal" or "double dispatch" mechanism in Kleckner (and as pointed out above, no need or purpose to be served by having such a mechanism). Since Klecker's processing is internal to the instrument, whose structure is readily available to its own internal methods, Kleckner does not

teach withholding knowledge of the overall referential event structure of the instrument from the processing mechanism. Again, this would be pointless in the context of Kleckner, and contrary to the way such processing is carried out in Kleckner. There is no reason or motivation apparent for combining those teachings of Kleckner with those of Rasala and/or Gould in regard to iterating and double dispatch, respectively, and the combined teachings do not teach the particular limitation addressed above that is recited in claim 17 (in short, the characteristic that the processing object lacks predetermined knowledge of financial instrument event structure).

Claim 18 reads as follows:

18. The system of claim 16, wherein a nested double dispatch mechanism initiated inside the action for a given financial event can select the appropriate action for any financial event referred to locally within the financial event.

Claim 18 recites a nested form of double dispatch, to deal with nested financial event references. Kleckner does not recite any circumstances in which there would be nested financial event references, and even if it had such references, they could be readily processed with a single dispatch mechanism, since the entire event structure is known in advance to its processing elements. Neither Kleckner nor the other cited references in combination with Kleckner teach nested double dispatch processing, nor is any suggesting or motivation apparent from these references to apply nested double dispatch to financial instrument processing.

Claim 19 reads as follows:

19. The system of claim 18 wherein said nested double dispatch mechanism can be applied recursively to any level.

Claim 19 adds recursion to the nested double dispatch of claim 18, so as to recursively nest the double dispatch mechanism to any level. The depth of the recursion corresponds to the depth to which financial event references are nested in the instrument being processed. Nothing of the sort is taught or suggested by Kleckner, Rasala or Gould, alone or in combination. Merely adding double dispatch as in Gould to a reference (Kleckner) that does not need or have any use for it does not obviate using a double dispatch mechanism to perform processing on a financial

instrument with an external processing object (as in the present invention), let alone nested double dispatch or recursively nested double dispatch.

CONCLUSION

The Applicant respectfully requests that the application be reconsidered in light of the present amendments and remarks, and that claims 11-19 be allowed.

Dated: November 7, 2001

Respectfully submitted,

Ronald Abramson
Registration No. 34,762
Attorney for Applicant

HUGHES HUBBARD & REED LLP
One Battery Park Plaza
New York, New York 10004-1482
(212) 837-6000